

Damien LU



Rapport de stage

*Étude de modèles de durée de vie,
application en fiabilité*

Stage du 08 juin 2020 au 07 août 2020

Structure d'accueil

le **cnam** Paris

Département EPN06 Mathématiques Statistiques

Tuteur Dariush GHORBANZADEH

Table des matières

1. Préambule	4
2. Contexte du stage.....	5
2.1. Organisme d'accueil.....	5
2.2. Sujet et objectifs du stage.....	5
2.3. Langages utilisés.....	6
2.4. Organisation et environnement de travail.....	6
3. Passerelle Python/PHP	9
3.1. Objectifs.....	9
3.2. Mise en place.....	9
3.3. Histogrammes.....	10
4. Modèles de durée de vie	11
4.1. Objectifs.....	11
4.2. Lois usuelles.....	11
4.3. Espérances, variances, moments d'ordre k.....	12
4.4. Fonction de répartition.....	13
4.5. Skewness et Kurtosis.....	13
4.6. Densité.....	14
5. Fichier de données.....	16
5.1. Objectifs.....	16
5.2. Fichier texte.....	16
5.3. Estimation de paramètres.....	17
5.4. Tests statistiques.....	18
5.4.1. Test du Khi-2.....	18
5.4.2. Test de Kolgomorov-Smirnov.....	19
6. Développement du site.....	20
6.1. Hébergement du site.....	20
6.2. Organisation des fichiers.....	21
6.3. Écriture du code.....	22
6.4. « Sécurité » du site web.....	23
Conclusion.....	24
Bibliographie.....	25
Annexe A environnement de travail.....	26
Annexe B développement durable.....	27

1. | Préambule

Remerciements

Malgré le contexte difficile, je tiens à remercier mon tuteur de stage Dariush Ghorbanzadeh de m'avoir accepté et de m'avoir encadré durant toute la durée du stage, en étant très disponible par mail et visioconférence. Merci également à lui d'avoir su me proposer des pistes d'amélioration de mon code.

Je remercie également mes camarades de stage avec qui nous avons pu progresser ensemble en développant en commun les parties communes à mon stage.

Je remercie enfin Dimitri Watel d'avoir été mon tuteur de stage à l'ENSIIE.

Résumé

Mon stage s'est déroulé au CNAM (conservatoire national des arts et métiers) en télétravail, dans le cadre de l'étude des modèles de durée de vie. Il s'agit de réaliser un site internet qui permet de traiter et d'étudier des données, en rapport avec les modèles de durée de vie : statistiques descriptives, étude de ces modèles, adéquation aux données et tests statistiques.

Mots-clés

traitement de données, fiabilité, tests, adéquation

2. Contexte du stage

2.1. Organisme d'accueil

Situé en plein coeur du Marais à Paris, le Conservatoire National des Arts et Métiers (CNAM) est un établissement d'enseignement supérieur et de recherche, possédant également le statut de grande école. Il permet d'accéder à plus de 500 formations diplômantes allant jusqu'au doctorat. Ces formations sont divisées en *écoles pédagogiques nationales* (EPN).



Figure 1 : Le CNAM Paris

Mon stage s'est déroulé dans le département Mathématiques et Statistiques (EPN06) : il assure des formations diplômantes du type LMD et diplômes d'ingénieur. L'enseignement est principalement dispensé sur des créneaux du soir (appelés HTO - *hors temps ouvrable*).

2.2. Sujet et objectifs du stage

Étude de modèles de durée de vie, application en fiabilité

L'objectif principal du stage est de créer une bibliothèque implémentée sous forme de site web qui modélise la théorie de la durée de vie, et qui permet :

- d'effectuer à partir d'un fichier de données une analyse statistique descriptive
- de consulter les différentes lois usuelles modélisant la théorie de la durée de vie
- à partir d'un fichier de données et d'une loi parmi les lois usuelles d'estimer ses paramètres par maximum de vraisemblance
- de tester l'adéquation des données à une des lois usuelles

La mise en place de ce site web sur un serveur a également été un aspect intéressant et enrichissant du stage même s'il n'a pas été prévu au départ.

Le travail a ainsi été réparti en cinq grandes phases :

- **Conception générale** : en commun avec les autres stagiaires de mon tuteur, développement d'une passerelle entre le site web et le traitement des données
- **Étude mathématique** du modèle de durée de vie et développement des scripts de traitement des données
- Implémentation de l'estimation des paramètres et des tests statistiques
- Mise en place de l'hébergement du site web
- Maquette du site et développement du site web

2.3. Langages utilisés

Le tuteur exige un traitement des données sous le langage Python. Ainsi, le site web tournera essentiellement avec le langage PHP, et non HTML, car il permet de définir et d'exécuter des scripts à la volée, et comportera des passerelles avec le langage Python pour traiter les données statistiques.

J'ai fait le choix d'un site web ne comportant que des pages PHP, car ce dernier pouvant lire indifféremment du code PHP ou HTML, il me permet d'être plus flexible lors du développement.

2.4. Organisation et environnement de travail

Compte tenu du contexte lié à l'épidémie de Covid-19, et à la fermeture temporaire du CNAM, mon stage s'est déroulé en télétravail. Pour cela, il a fallu s'organiser avec mon tuteur de stage : ainsi, avec les autres stagiaires sous sa responsabilité, ce dernier a mis en place une plateforme Microsoft Teams (figure 2) qui permet de collaborer en équipe et d'organiser des séances en visioconférence pour partager le travail commun à tous les stagiaires et également pour discuter avec le tuteur de stage :

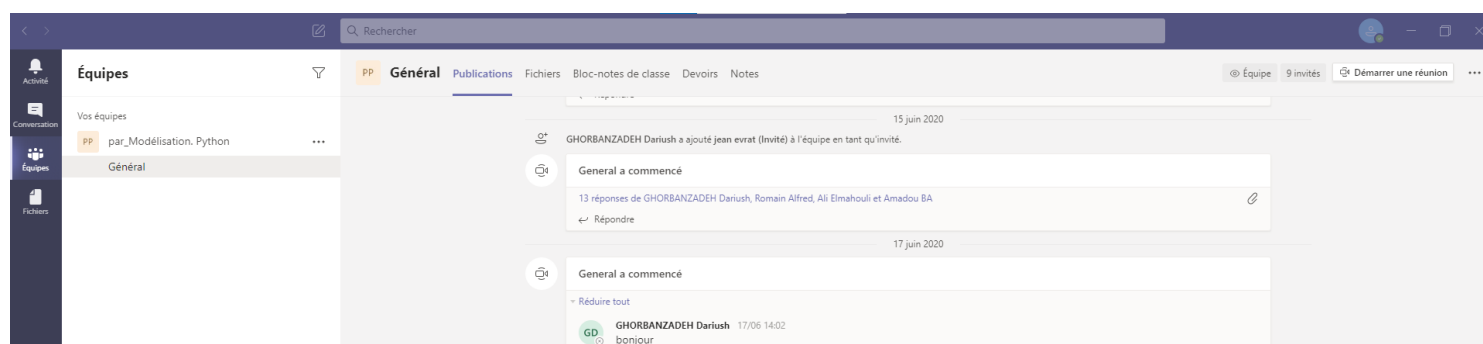


Figure 2 : Interface de Microsoft Teams pour la collaboration

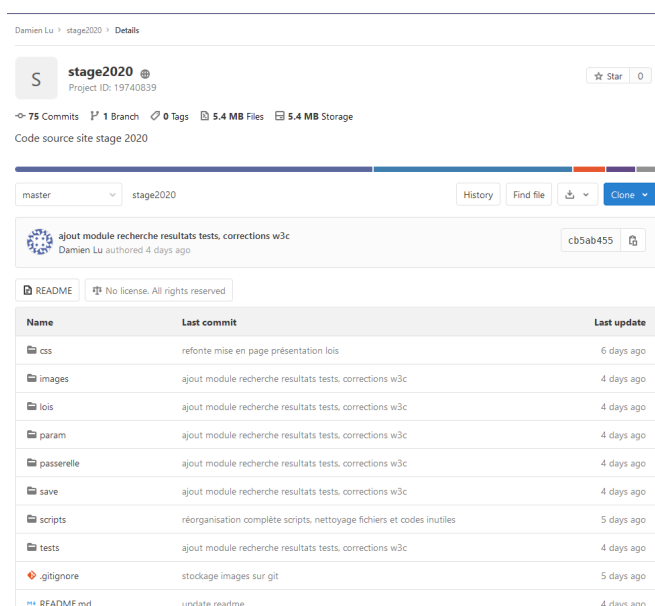


Figure 3 : Dépôt Gitlab du stage

Lorsque le code a été suffisamment mature, j'ai également mis en place un dépôt Gitlab (figure 3) à l'adresse ci-contre : <https://gitlab.com/dlu02/stage2020> pour que mon tuteur de stage puisse accéder au code source du site web et le vérifier. Les droits d'accès ont été mis en *public* pour que ce dernier puisse y accéder facilement sans lui imposer de créer un compte Gitlab.

Par ailleurs, le développement d'un site web étant bien plus commode en observant en temps réel le résultat de la page, et la communication avec mon tuteur de stage se faisant à distance, j'ai mis en place un serveur Web à mon domicile qui héberge le site web créé (figure 4) à l'adresse ci-contre : <http://dlu02.freeboxos.fr/stage>. Ainsi, mon tuteur de stage peut accéder en temps réel à mon travail et peut me faire part de corrections : en effet, la mise à jour du dépôt Gitlab entraîne automatiquement celle du site web hébergé, car sur le serveur, le serveur web Apache lit les données directement dans le dépôt Git.

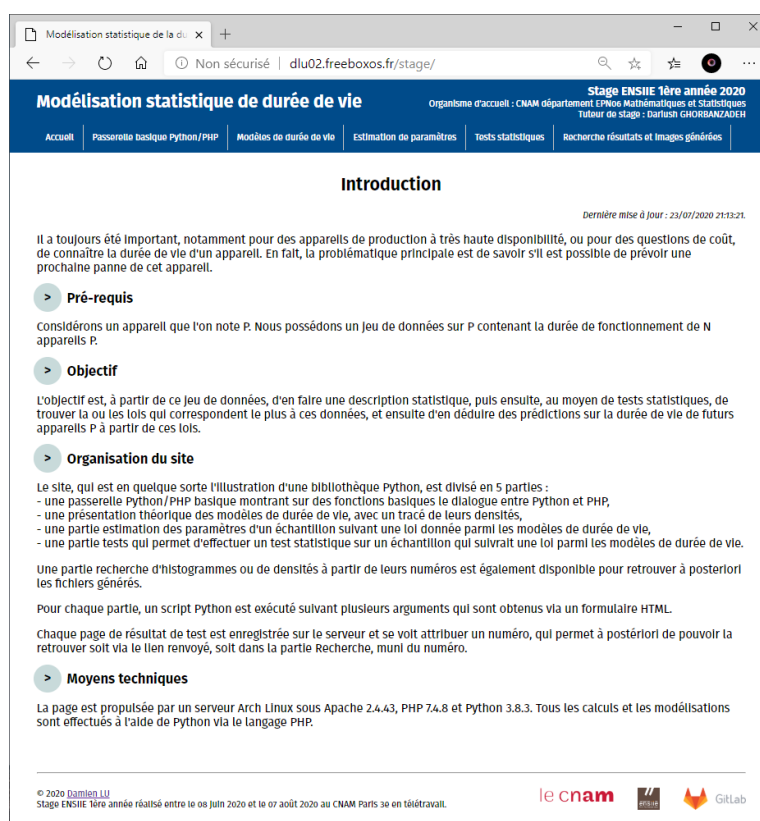


Figure 4 : Aperçu du site web

Enfin, pour éditer directement les fichiers du serveur, j'ai mis en place sur le serveur un accès SSH et SFTP par le nom de domaine du site (figure 5), qui permet de monter sous Linux directement le répertoire du serveur web dans l'environnement du bureau, et qui permet de travailler de n'importe où, ce qui permet de corriger des erreurs urgentes directement sur mon smartphone par exemple. J'ai également changé le port SSH pour plus de sécurité et permettre d'éviter les attaques ciblées sur le port standard SSH (port 22).

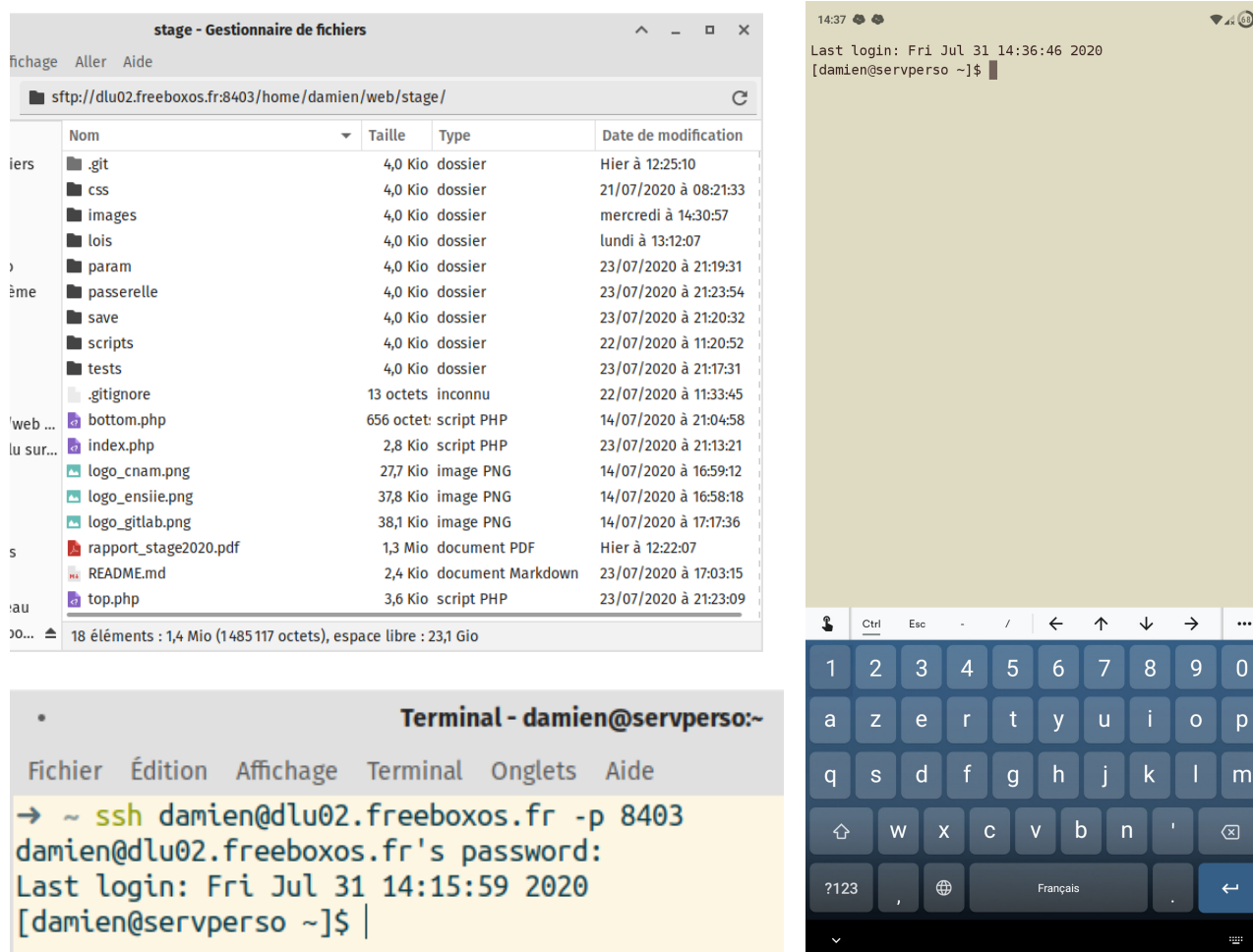


Figure 5 : connexions SFTP et SSH sur PC et Android

Le répertoire distant étant monté, il suffit alors de l'ouvrir avec mon éditeur de texte Atom, et on obtient mon espace de travail (voir [Figure A](#) dans l'annexe A), contenant le contenu du répertoire, les erreurs eventuelles du fichier et un menu Git permettant de commiter directement sans ligne de commande.

3. Passerelle Python/PHP

3.1. Objectifs

La première partie du stage a été de prendre en main la passerelle entre Python et PHP. Elle a duré environ deux semaines, et a consisté à écrire plusieurs fonctions basiques Python, qui allaient nous servir dans la suite du stage, et les implémenter dans une page PHP :

- recherche d'un fichier dans l'arborescence du site
- générateur de nombres aléatoires en Python
- prise en main du module Argparse qui permet l'exécution d'un script Python muni d'un ou plusieurs arguments.
- tracé d'histogrammes et de densités

Ce travail est commun aux autres stagiaires, et nous avons pu lors de conférences Teams mettre en commun nos travaux et avoir un retour du tuteur de stage.

3.2. Mise en place

Les pages de passerelle obéissent toutes à une même structure : elles ne sont que des pages PHP classiques auxquelles j'ajoute une commande Python qui exécute le script muni de ses arguments : par exemple, le script PHP

php

```
$resultat = shell_exec("python gen_aleat.py $min $max $taille $fichier");
```

exécute Python qui lui exécute le script `gen_aleat.py` muni des arguments `$min $max $taille $fichier` et met le résultat dans la variable `$resultat`.

Ainsi, la page `passerelle.php` affichant le générateur de nombres aléatoires comporte un formulaire HTML qui demande à l'utilisateur le minimum, le maximum, le nombre d'éléments à générer et le nom du fichier de sortie, et renvoie à une page PHP `gen_aleat.php` de traitement ces arguments via la méthode POST, ce qui permet de ne pas avoir les arguments présents dans le lien de la page.

Bien sûr, il est absolument nécessaire de vérifier la pertinence des données saisies par l'utilisateur : c'est pourquoi dès le formulaire, j'indique que les champs minimum, maximum et nombre d'éléments à générer sont des entiers, ce qui rend impossible toute saisie de chaîne de caractères, puis je vérifie que la saisie n'est pas vide.

Pour une page de saisie `saisie.php`, le schéma de traitement via Argparse est donc représenté dans la figure 6 :

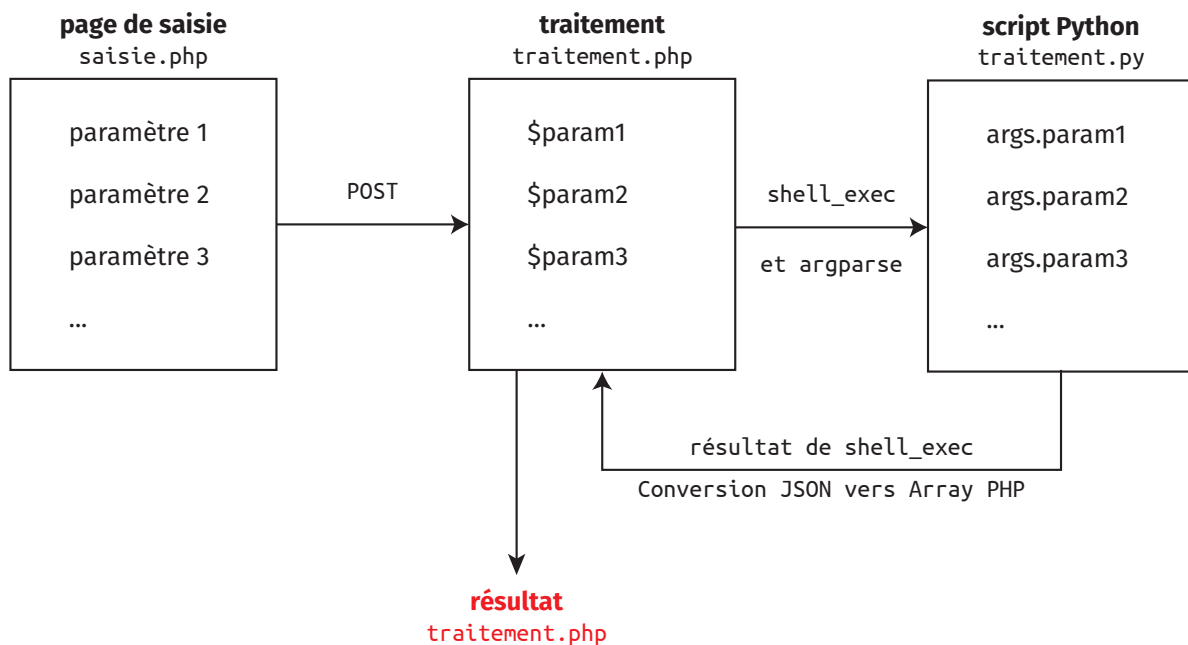


Figure 6 : traitement des scripts Python

Dans le cas où le script Python renvoie plusieurs valeurs, il faut alors les mettre dans un dictionnaire, puis le module `json` de Python convertit ce dictionnaire en un dictionnaire JSON. Dans PHP, il suffit alors de convertir ce dictionnaire JSON en un Array PHP par la fonction `json_decode`. On peut alors accéder à chaque case de l'array via la clé du dictionnaire.

3.3. Histogrammes

Le module Matplotlib de Python propose un tracé d'histogramme à partir d'un tableau de données Numpy, mais cet histogramme ne va pas pour des données continues. Ainsi, mon tuteur de stage m'a fourni une fonction de tracé d'histogrammes correcte à partir d'un nombre de classes correct. Cette fonction est contenue dans le fichier `histogramme.py` et est importé si nécessaire dans les autres scripts Python via la syntaxe :

```
python> from histogramme import histo_Continue
```

Cette fonction a besoin du nombre de classes n_c de l'échantillon de taille N . Je le détermine avec la formule (1) :

$$n_c = 5 \times \log_{10}(N) \quad (1)$$

4. Modèles de durée de vie

4.1. Objectifs

La deuxième partie du stage est consacrée à l'étude théorique des modèles de durée de vie. Il s'agit, à partir de lois usuelles données par mon tuteur de stage, de déterminer les expressions théoriques de l'espérance, de la variance, des moments d'ordre k , du skewness et du kurtosis. Je présente ensuite les résultats sur une page PHP dans la rubrique *Modèles de durée de vie*. Chaque loi possède sa page de présentation. J'ai également implémenté un module de tracé de la densité en fonction des paramètres de chaque loi.

Cette partie du stage a demandé environ 2 semaines, en comptant les vérifications des calculs et du code.

4.2. Lois usuelles

La théorie de la durée de vie repose sur des lois usuelles qui permettent de modéliser ce phénomène. Mon tuteur de stage a retenu six lois qui sont les suivantes :

- **Loi hyperexponentielle** : paramètres $a > 0$ et $b > 0$:

$$f(x, a, b) = \frac{ab}{a+b} \left(e^{-ax} + e^{-bx} \right) 1_{]0, +\infty[}(x) \quad (2)$$

- **Loi Lomax** : paramètres $a > 0$ et $b > 0$:

$$f(x, a, b) = \frac{ab^a}{(b+x)^{a+1}} 1_{]0, +\infty[}(x) \quad (3)$$

- **Loi de Weibull** : paramètres $a > 0$ et $b > 0$:

$$f(x, a, b) = \frac{bx^{b-1}}{a^b} e^{-(x/a)^b} 1_{]0, +\infty[}(x) \quad (4)$$

- **Loi exponentielle polynomiale** : paramètres $a = (a_1, \dots, a_m) \geq 0$, $b > 0$ et $m \in \mathbf{N}^*$:

$$f(x, a, b) = C(a, b) \left(\sum_{k=1}^m a_k x^k \right) e^{-bx} 1_{]0, +\infty[}(x) \quad (5)$$

où $C(a, b)$ est tel que $\int_{-\infty}^{+\infty} f(x, a, b) dx = 1$ et vaut : $C(a, b) = \left(\sum_{k=1}^m a_k \frac{k!}{b^{k+1}} \right)^{-1}$

- **Loi de Burr** : paramètres $a > 0$ et $b > 0$:

$$f(x, a, b) = \frac{abx^{a-1}}{(1+x^a)^{b+1}} 1_{]0, +\infty[}(x) \quad (6)$$

— **Loi exponentielle convolution** : paramètres $a > 0$ et $b > 0$:

$$f(x, a, b) = \begin{cases} \frac{ab}{b-a} (e^{-ax} - e^{-bx}) 1_{]0, +\infty[}(x) & \text{si } a \neq b \\ a^2 x e^{-ax} & \text{si } a = b \end{cases} \quad (7)$$

4.3. Espérances, variances, moments d'ordre k

Je détermine ensuite les espérances, variances et moments d'ordre k des lois usuelles. Pour cela, j'utilise la formule du transfert (8), et la formule de Koenig-Huygens (9) :

$$\mathbf{E}(X^k) = \int_{-\infty}^{+\infty} x^k f(x, a, b) dx \quad (8)$$

$$\mathbf{V}(X) = \mathbf{E}(X^2) - \mathbf{E}(X)^2 \quad (9)$$

À titre d'exemple, effectuons le calcul pour la loi hyperexponentielle :

$$\begin{aligned} \mathbf{E}(X^k) &= \int_0^{+\infty} \frac{ab}{a+b} (e^{-ax} + e^{-bx}) dx \\ &= \frac{b}{a+b} \int_0^{+\infty} x^k a e^{-ax} dx + \frac{a}{a+b} \int_0^{+\infty} x^k b e^{-bx} dx \\ &= \frac{b}{a+b} \frac{k!}{a^k} + \frac{a}{a+b} \frac{k!}{b^k} \quad \text{après } k \text{ intégrations par parties} \end{aligned}$$

$$\begin{aligned} \mathbf{V}(X) &= \mathbf{E}(X^2) - \mathbf{E}(X)^2 \\ &= \frac{2b}{a^2(a+b)} + \frac{2a}{b^2(a+b)} - \frac{b^2}{a^2(a+b)^2} - \frac{a^2}{b^2(a+b)^2} \\ &= \frac{b^2 + 2ab}{a^2(a+b)^2} + \frac{a^2 + 2ab}{b^2(a+b)^2} \end{aligned}$$

Quant aux autres lois usuelles, je ne vais pas détailler ici les calculs, car ce serait beaucoup trop long, mais le procédé est similaire, seul le calcul de l'intégrale diffère suivant les lois : en général j'effectue soit une ou plusieurs intégrations par parties, soit un changement de variable, et ensuite pour certaines lois, je reconnais la fonction Gamma d'Euler.

Les résultats obtenus pour les autres lois usuelles sont consultables sur le site web.

4.4. Fonction de répartition

Par définition de la fonction de répartition, on a, pour toute variable aléatoire Y continue et de densité $f(x, a, b)$:

$$F_Y(x) = P(Y \leq x) = \int_{-\infty}^x f(x, a, b) dx \quad (10)$$

Ceci donne pour la loi hyperexponentielle :

$$\begin{aligned} F_X(x) &= \int_{-\infty}^x f(x, a, b) dx \\ &= \int_0^x \frac{ab}{a+b} (e^{-ax} + e^{-bx}) dx \\ &= \frac{ab}{a+b} \left[\frac{e^{-ax}}{-a} + \frac{e^{-bx}}{-b} \right]_0^x \\ &= \frac{ab}{a+b} \left(\frac{1 - e^{-ax}}{a} + \frac{1 - e^{-bx}}{b} \right) \end{aligned}$$

Pour les autres lois usuelles, le principe est le même, seul le calcul de l'intégrale change.

4.5. Skewness et Kurtosis

Pour définir les notions de skewness et kurtosis, j'ai d'abord besoin de la notion de moment centré d'ordre k , que je définis ainsi :

Définition

Soit X une variable aléatoire continue, admettant une espérance notée M . On définit le **moment centré d'ordre k** et on note μ_k la quantité :

$$\mu_k = \mathbf{E}((X - M)^k) = \int_{-\infty}^{+\infty} (x - M)^k f(x, a, b) dx \quad (11)$$

Ainsi, on peut définir le skewness et le kurtosis, notés respectivement γ_1 et γ_2 :

$$\gamma_1 = \frac{\mu_3}{\mu_2^{3/2}} \quad \gamma_2 = \frac{\mu_4}{\mu_2^2} \quad (12)$$

Or, le calcul des moments centrés est difficile. De par la définition du moment centré, on peut tirer une expression du skewness et du kurtosis qui ne fait intervenir que les moments d'ordre k , l'espérance et la variance : en effet, on a, en notant M l'espérance :

$$\mu_k = \mathbf{E}((X - M)^k) = \mathbf{E}\left(\sum_{i=0}^k \binom{k}{i} X^i (-1)^i M^{k-i}\right) = \sum_{i=0}^k \binom{k}{i} (-1)^i M^{k-i} \mathbf{E}(X^i) \quad (13)$$

donc :

$$\begin{aligned} \mu_1 &= \mathbf{E}(X - M) & \mu_2 &= \mathbf{E}((X - M)^2) & \mu_3 &= \mathbf{E}((X - M)^3) \\ &= \mathbf{E}(X) - M & &= \mathbf{E}(X^2) - 2M\mathbf{E}(X) + M^2 & &= \mathbf{E}(X^3) - 3M\mathbf{E}(X^2) + 3M^2\mathbf{E}(X) - M^3 \\ & & &= \mathbf{E}(X^2) - \mathbf{E}(X)^2 & &= \mathbf{E}(X^3) - 3M\mathbf{E}(X^2) + 2M^3 \\ & & &= \mathbf{V}(X) & & \end{aligned}$$

$$\begin{aligned} \mu_4 &= \mathbf{E}((X - M)^4) \\ &= \mathbf{E}(X^4) - 4M\mathbf{E}(X^3) + 6M^2\mathbf{E}(X^2) - 4M^3\mathbf{E}(X) + M^4 \\ &= \mathbf{E}(X^4) - 4M\mathbf{E}(X^3) + 6M^2\mathbf{E}(X^2) - 3M^4 \end{aligned}$$

et on obtient les expressions finales des skewness et kurtosis, toujours en notant M l'espérance :

$$\gamma_1 = \frac{\mathbf{E}(X^3) - 3M\mathbf{E}(X^2) + 2M^3}{\mathbf{V}(X)^{3/2}} \quad \gamma_2 = \frac{\mathbf{E}(X^4) - 4M\mathbf{E}(X^3) + 6M^2\mathbf{E}(X^2) - 3M^4}{\mathbf{V}(X)^2} \quad (14)$$

Le fichier `lois_theo.py` contient toutes les expressions théoriques des grandeurs de cette partie, et elles peuvent être réutilisées par importation si nécessaire.

4.6. Densité

Après avoir déterminé les expressions théoriques des principales grandeurs statistiques, il faut maintenant donner une représentation graphique de la loi. Pour cela, la clé est de représenter la partie « intéressante » du graphique, en effet, pour tous les modèles de durée de vie, la densité tend en décroissant vers 0 en l'infini. Autrement dit, je dois trouver la valeur de x qui permette d'avoir un pourcentage P suffisamment proche de 1 des valeurs de la densité, ce qui revient à résoudre en x

$$F_Y(x, a, b) = P$$

où F_Y est la fonction de répartition de la variable aléatoire Y .

Pour les lois du modèle, il est très difficile voire impossible de résoudre analytiquement cette équation. Pour y remédier, je vais faire appel à la fonction `fsolve` du module `Scipy` de `Python` qui va résoudre cette équation numériquement, avec un pourcentage P fixé à l'avance à $P = 0,98$.

Ainsi, j'ai donc écrit une bibliothèque `densite.py` qui contient notamment la fonction de tracé de la densité d'une loi du modèle selon ses paramètres. Sur chaque page présentant une loi du modèle, un formulaire est présent en plus des valeurs théoriques des données statistiques pour tracer la densité en fonction de ses paramètres. J'applique alors

le même procédé que pour la passerelle et j'écris un script Python qui prend en argument les paramètres de la loi et la loi à tracer et retourne sous forme de dictionnaire à une valeur le numéro de la densité tracée.

Concernant le cas particulier de l'exponentielle polynomiale, étant donné qu'un des deux paramètres est un n-uplet de nombres positifs, le formulaire demande l'écriture du n-uplet selon la syntaxe CSV, ie. avec une séparation en espaces ou avec une virgule, de sorte que le script PHP envoie cet n-uplet dans un fichier temporaire temp.txt, qui sera lu par Python (via le nom du fichier passé en argument), comme résumé dans la figure 7 :

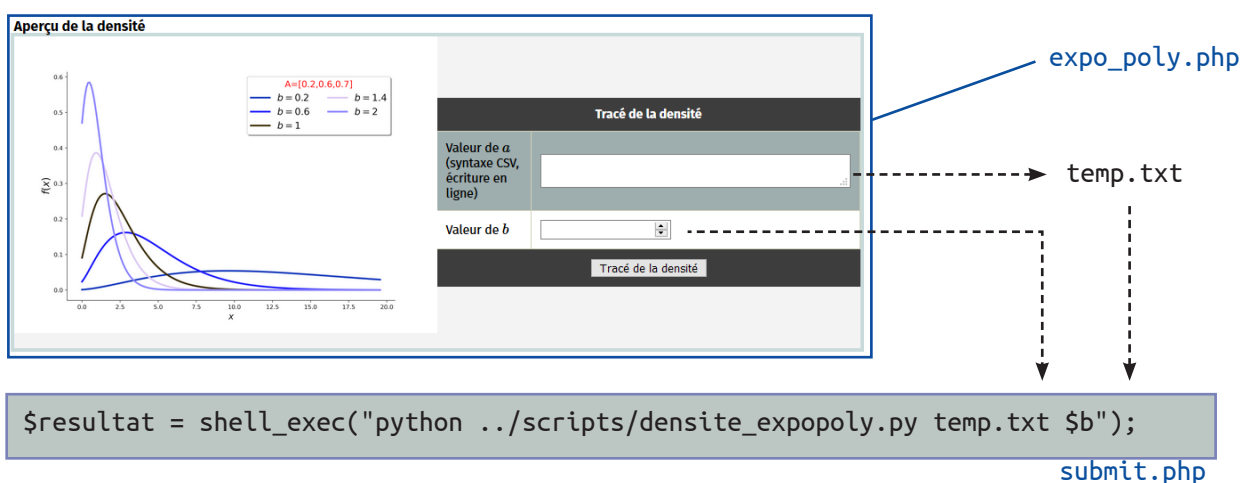


Figure 7 : traitement du tracé de la densité de l'exponentielle polynomiale

5. Fichier de données

5.1. Objectifs

Maintenant que les modèles ont été décrits théoriquement, je m'intéresse ensuite à l'étude pratique. L'objectif est donc, à partir d'un fichier de données et d'une loi parmi les modèles :

- de faire une description statistique des données,
- d'estimer ses paramètres par maximum de vraisemblance,
- de tester l'adéquation à la loi, en implémentant le test du Khi-2 et de Kolgomorov-Smirnov

J'ai également implémenté une fonction d'enregistrement des résultats : pour chaque test d'adéquation effectué, un numéro est renvoyé, la page est enregistrée sous ce numéro, et peut être retrouvée avec ce numéro.

5.2. Fichier texte

Tout d'abord, il faut préciser ce qu'est un fichier texte. Je limite, par choix, les fichiers textes aux extensions *.txt, *.dat et *.csv. Toute autre extension sera refusée par le site. En effet, le code deviendrait vite lourd s'il faut distinguer toutes les extensions possibles.

Ensuite, le site permet l'étude d'une variable aléatoire, donc d'un seul type de données. J'ai donc fait le choix de transformer le fichier envoyé au site. Tout séparateur CSV (espace, virgule, point virgule) sera remplacé par un saut de ligne, de sorte à avoir finalement un fichier contenant une seule colonne de données. Cette transformation est effectuée avant l'envoi du fichier à Python par un appel système via la commande sed. La figure 8 montre ce processus de transformation :

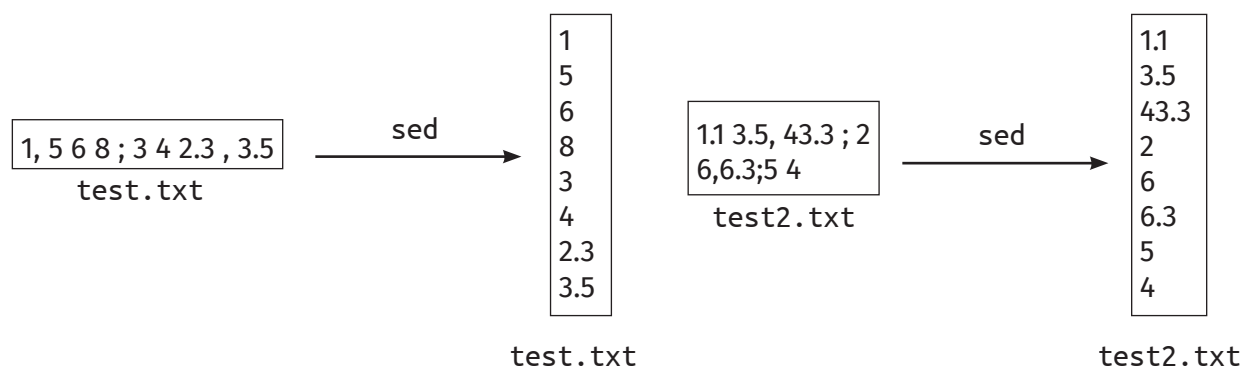


Figure 8 : transformation du fichier texte par sed

5.3. Estimation de paramètres

Pour estimer les paramètres, je vais utiliser le modèle du maximum de vraisemblance, qui est plus précise asymptotiquement que la méthode des moments. Mon tuteur de stage propose en outre d'ajouter des contraintes pour rendre l'estimation encore plus précise. On obtient finalement deux modèles :

- **Modèle 1** : contraintes sur l'espérance et le moment d'ordre 2 empiriques

$$\max_{a,b} \mathcal{L}(x_1, \dots, x_n; a, b) \quad \text{s.c.} \quad \begin{cases} \mathbf{E}(X) = \frac{1}{n} \sum_{i=1}^n x_i \\ \mathbf{E}(X^2) = \frac{1}{n} \sum_{i=1}^n x_i^2 \end{cases} \quad (15)$$

- **Modèle 2** : contrainte sur le signe du Skewness

$$\max_{a,b} \mathcal{L}(x_1, \dots, x_n; a, b) \quad \text{s.c.} \quad \mathbf{E}((X - \mathbf{E}(X))^3) \gamma_1 \geq 0 \quad (16)$$

Pour chaque modèle, il s'agit alors de résoudre un problème d'optimisation sous contraintes, et il est très difficile d'obtenir une solution analytique à ce système. C'est pour cela que je vais résoudre numériquement ce système à l'aide de la fonction `optimize` du module `scipy` de Python. Comme pour la fonction `fsolve` du même module, cette fonction demande un point de départ. Je fais alors le choix de prendre un point aléatoire et de vérifier ensuite si la solution obtenue est bien adaptée, et j'obtiens l'algorithme de la figure 9 :

```

Pour k allant de 1 à 10, faire
  Choisir un point aléatoire
  Résoudre le système avec optimize
  Si gradient(logvs, point) < 1e-6, alors
    Si matrice_hessienne est définie positive alors
      Retourner le point
Retourner le point pour lequel le gradient est le plus proche de 0

```

Figure 9 : algorithme de résolution du problème

Notons que les fonctions gradient et hessienne ont été implémentées sous Python sous forme numérique, par les formules de dérivée numérique, et que pour le modèle 2, la contrainte fait que nous nous intéressons de fait à la forme de la distribution. Pour garantir un temps de calcul acceptable, j'ai fait le choix de limiter à 10 itérations l'algorithme, et de retourner dans le cas où aucun des points trouvés ne vérifie la condition du gradient le point qui s'en rapprocherait le plus.

La page de résultat affiche alors en plus de l'analyse statistique descriptive les valeurs de a et b déterminées par maximum de vraisemblance, suivant le même procédé décrit précédemment. Dans la figure 10 à la page suivante se trouve un aperçu des pages de présentation et de résultats.

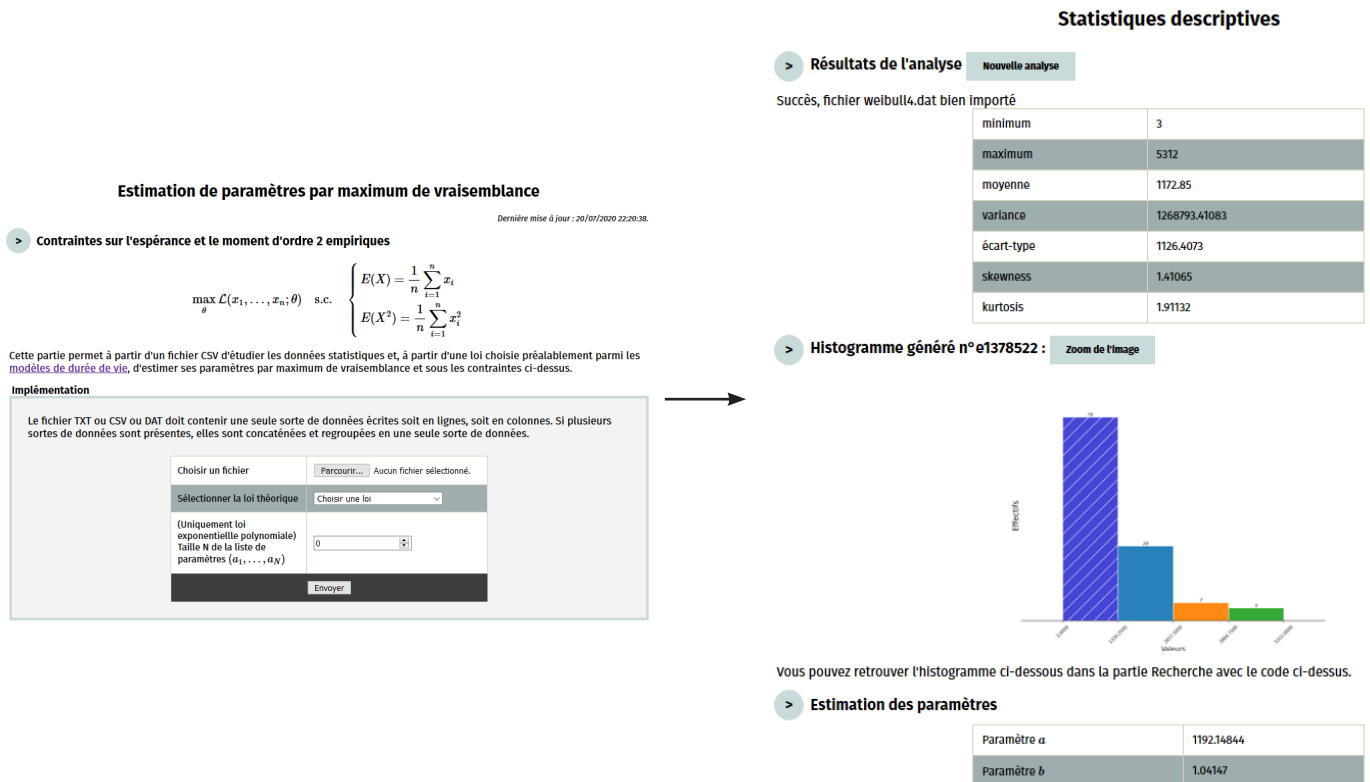


Figure 10 : aperçu de la page d'estimation

5.4. Tests statistiques

L'estimation des paramètres va me servir à implémenter les tests statistiques. En effet, il faut maintenant tester l'adéquation à cette loi avec les paramètres estimés. Pour cela, je vais implémenter deux tests statistiques : le test du Khi-2 et le test de Kolgomorov-Smirnov. Il existe des fonctions dans le module Scipy (`kstest` et `khi2`) qui implémentent directement ces tests mais je vais plutôt construire entièrement les tests, et donc les coder entièrement.

5.4.1. Test du Khi-2

Soit un fichier de données contenant N observations. Nous voulons tester l'adéquation à une loi L de paramètres estimés a et b, et de fonction de répartition fdr.

Séparons les données du fichier à étudier en n classes. Chaque classe a donc sa fréquence théorique et sa fréquence empirique d'observation. Le but du test du Khi-2 est de comparer ces deux distributions à l'aide de ces fréquences, à l'aide de cette distance suivante :

$$T = \sum_{i=1}^n \frac{(n_i - Np_i)^2}{Np_i} \quad (17)$$

où n_i est le nombre d'observations empirique de la classe C_i et Np_i est le nombre d'observations théorique de la classe C_i .

Cette distance correspond à la statistique de test, et elle suit une loi du Khi-2 à $n-1$ degrés de liberté. La valeur p du test s'obtient alors en prenant le « complémentaire » de la fonction de répartition du Khi-2 en T , et nous permet de conclure :

- si la valeur p est inférieure à 0.05, l'échantillon n'a aucune chance de suivre la loi L
- sinon, il y a une possibilité que l'échantillon suive la loi L .

Pour l'implémentation du test, j'ai procédé par étapes :

- **définition des classes** : fonction `int_classe`
- **nombre d'occurrences empirique par classe** : fonction `tableau`
- **fréquence empirique par classe** : fonction `freq_emp`
- **fréquence théorique par classe** : fonction `freq_theo`
- **calcul de la distance et de la p-valeur** : fonction `chi2`

Toutes ces fonctions sont réunies dans le fichier `tests.py` qui est le script Python d'exécution des tests.

5.4.2. Test de Kolgomorov-Smirnov

Toujours dans le même contexte, soit un fichier de données contenant N observations. Nous voulons tester l'adéquation à une loi L de paramètres estimés a et b , et de fonction de répartition F .

Le test de Kolgomorov-Smirnov consiste non pas à comparer les fréquences théoriques et empiriques, mais à comparer les fonctions de répartition théorique et empirique, à l'aide de la distance suivante :

$$D = \max(F_n(x) - F(x)) \quad (18)$$

D'après le théorème de Kolgomorov, que je vais admettre ici, sous l'hypothèse H_0 du test, pour tout $\lambda > 0$:

$$P\left(D > \frac{\lambda}{\sqrt{n}}\right) \xrightarrow{n \rightarrow +\infty} 2 \sum_{k=1}^{+\infty} (-1)^k \exp(-2k^2 \lambda^2) \quad (19)$$

et la convergence est dans ce cas extrêmement rapide. Par la définition de la valeur p :

$$P = 1 - F_0(T) \quad (20)$$

où F_0 est la fonction de répartition de la statistique de test sous l'hypothèse H_0 et T est la statistique du test, on obtient par le théorème de Kolgomorov la valeur p du test de Kolgomorov Smirnov qui nous permet de conclure par la même manière que pour le test du Khi-2.

Pour l'implémentation du test, j'ai procédé en deux étapes :

- **fonction de répartition empirique** : fonction `fdr_empirique`
- **calcul de la distance et de la p-valeur** : fonction `testks`

Toutes ces fonctions sont également réunies dans le fichier `tests.py` qui est le script Python d'exécution des tests.

6. Développement du site

6.1. Hébergement du site

Comme mon stage se déroule en télétravail, il fallait, pour compenser l'absence de suivi quotidien en présentiel, rendre l'accès à mon travail plus facile pour mon tuteur de stage. C'est pour cela que j'ai pensé à héberger le site web sur un serveur, disponible 24h/24. Possédant un ordinateur portable en réserve, je l'ai donc configuré sous une Arch Linux munie du serveur web Apache 2.4.43, de PHP 7.4.8 et de Python 3.8.5. Le choix d'Arch Linux se justifie par la volonté d'avoir rapidement les dernières versions des logiciels, mais en contrepartie, je sacrifie potentiellement un peu de stabilité, en cas de souci sur les dernières versions de logiciels.

Ensuite, il faut configurer l'accès distant au site web : j'ai donc effectué une redirection de port pour le serveur. Je redirige les flux entrants au port 80 vers ma machine d'adresse IP privée 192.168.1.15 au port 80.

Et enfin, pour éviter d'avoir à saisir mon adresse IP publique pour accéder au domaine, et puisque mon fournisseur d'accès à Internet me permet d'obtenir un nom de domaine gratuit à suffixe, je l'ai donc utilisé et configuré pour qu'il redirige vers mon IP publique. Un aperçu de la configuration de l'accès distant se trouve ci-contre figure 11.

Un des problèmes de ces noms de domaine gratuits est qu'ils sont impossibles à configurer manuellement. En effet, l'accès au site par HTTPS (port 443) est impossible faute de certificat disponible pour les machines personnelles, et les navigateurs web affichent donc une alerte dans la barre d'adresse. Cela ne pose pas de problème pour mon site actuel vu qu'il ne traite pas des données sensibles, mais un projet futur serait d'acquérir un nom de domaine personnalisé et de pouvoir le configurer pour rendre le HTTPS disponible.

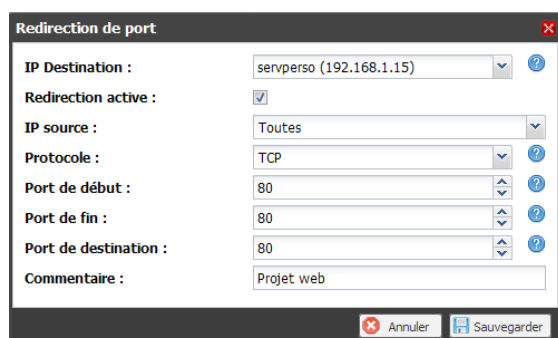


Figure 11 : serveur hébergeant le site web et configuration réseau

6.2. Organisation des fichiers

J'ai choisi de ne pas implémenter le modèle MVC pour le site. En effet, j'ai préféré une organisation personnelle qui est plus facile à comprendre au vu de la taille modeste du site.

Notons / le répertoire racine que lit le serveur web. Dans / se trouve la page d'accueil, nommée `index.php`. J'ai choisi d'écrire un header et un bottom dans des fichiers séparés `top.php` et `bottom.php` pour utiliser la possibilité d'`include` des fichiers dans PHP. Ces fichiers contiennent l'en-tête (menu supérieur) et le pied de la page et sont également situés dans /.

Chaque partie de la page web possède son propre dossier : passerelle, param pour l'estimation, tests, lois (pour les modèles de durée de vie). Tous les scripts Python appelés par les pages PHP sont situés dans `/scripts`. Ainsi, par exemple pour le générateur de nombres aléatoires situé dans `/passerelle`, l'appel à Python devient :

php

```
$resultat = shell_exec("python ../scripts/gen_aleat.py $min $max $taille $fichier");
```

Ainsi, tous les appels aux scripts Python se font par l'écriture du chemin relatif du script, et de même, l'importation des menus, du pied de page PHP (`top.php` et `bottom.php`), ainsi que de la feuille de style CSS se fait par chemin relatif de ces fichiers. Seuls les liens vers les pages du site contenus dans le menu sont exprimés par chemin absolu, car le menu est censé être importé dans toutes les pages du site, donc à des répertoires différents.

Un aperçu de l'arborescence du projet se trouve figure 12.

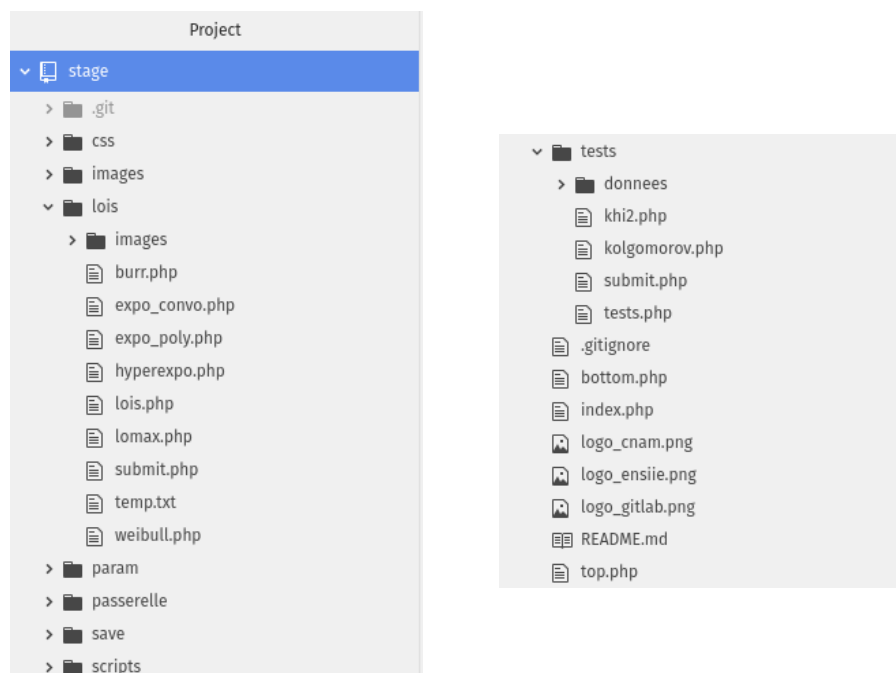


Figure 12 : arborescence du projet

6.3. Écriture du code

Au vu de la souplesse du langage PHP, j'ai fait le choix d'écrire dans chaque fichier le code en HTML comme si la page était en HTML et d'exécuter des requêtes PHP uniquement lorsque c'est nécessaire en les séparant par les balises `<?php ?>`.

Lorsque la page visitée par l'utilisateur comporte un formulaire, elle transmet ensuite les données saisies par l'utilisateur à une page annexe de traitement située dans le même répertoire. Elle comporte la même mise en page que les autres pages et affiche à l'utilisateur le résultat des traitements et requêtes PHP exécutées. Pour les besoins du tuteur de stage, chaque page comporte et affiche sa date de modification.

J'ai également ajouté trois scripts Javascript : ils permettent l'importation de la police Fira Sans depuis Google Font pour permettre l'homogénéisation de la mise en page — ce rapport est également écrit en police Fira Sans —, la possibilité d'écrire et d'afficher à l'utilisateur du code LaTeX pour améliorer l'écriture des formules mathématiques, et la conservation du menu supérieur lors du scrolling vers le bas. Ces scripts sont situés dans l'en-tête, de sorte qu'ils puissent être disponibles pour chaque page, puisque l'en-tête est importé pour chaque page du site. J'ai fait en sorte de réduire au maximum les scripts pour rendre les pages plus légères.

L'écriture du code a pu se faire de façon facilitée grâce à mon éditeur de texte, Atom : en effet, on constate par la figure 13 qu'il signale par des outils d'analyse syntaxique (php -l et flake8 pour Python) en direct toute erreur de syntaxe par un point rouge et pour flake8 me force en plus à rendre mon code Python conforme à la convention PEP 8 (Python Enhancement Proposals, 2001, destinée à améliorer la lisibilité et la compréhension du code).

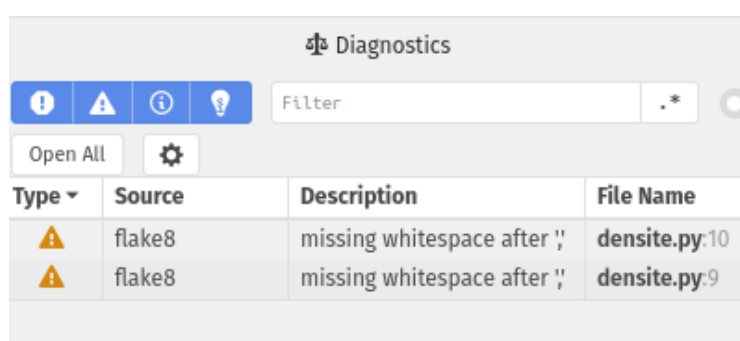
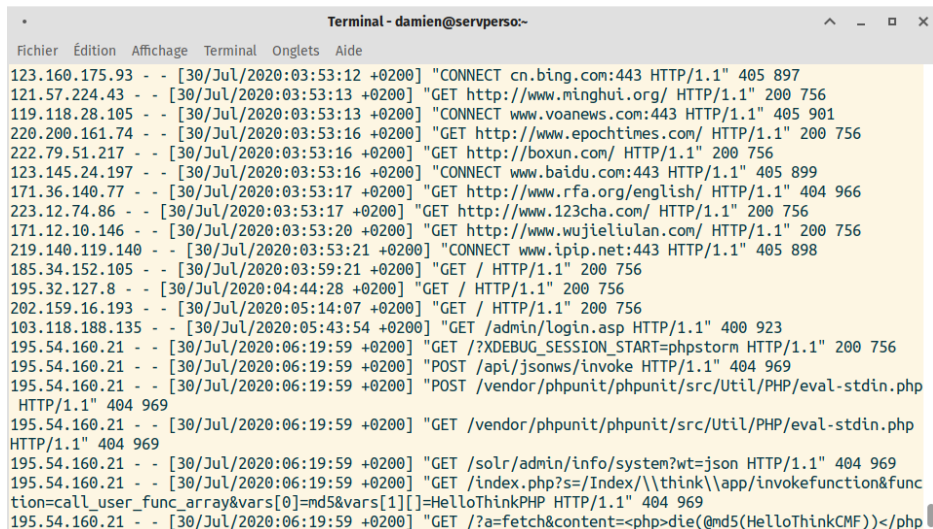


Figure 13 : éditeur Atom affichant des warning de mise en forme

6.4. « Sécurité » du site web

Étant donné que le site est hébergé, l'aspect sécurité du site est important. En effet, les logs du serveur web de la figure 14 montrent d'incessantes tentatives d'intrusion du serveur par des robots ou des personnes. De même, vu que le site nécessite d'envoyer des fichiers au serveur, il faut vérifier et s'assurer de la pertinence des données collectées.

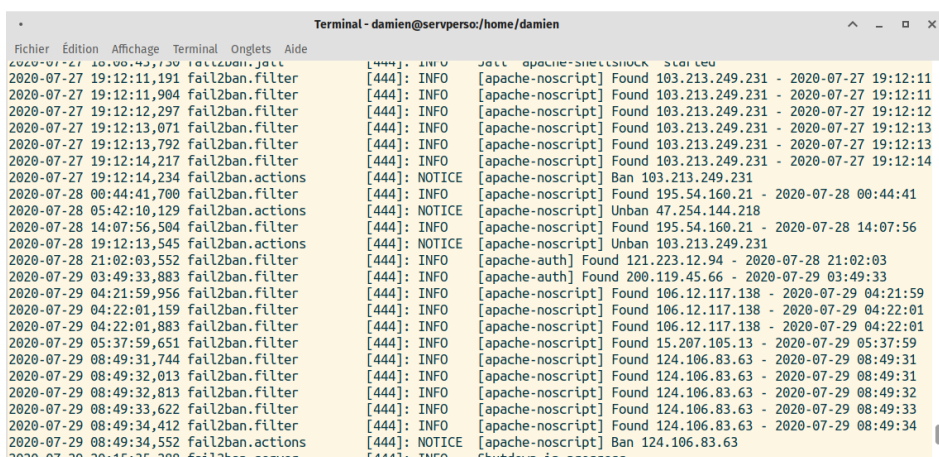


```

Terminal - damien@servperso:~
Fichier Édition Affichage Terminal Onglets Aide
123.160.175.93 - - [30/Jul/2020:03:53:12 +0200] "CONNECT cn.bing.com:443 HTTP/1.1" 405 897
121.57.224.43 - - [30/Jul/2020:03:53:13 +0200] "GET http://www.minghui.org/ HTTP/1.1" 200 756
119.118.28.105 - - [30/Jul/2020:03:53:13 +0200] "CONNECT www.voanews.com:443 HTTP/1.1" 405 901
220.200.161.74 - - [30/Jul/2020:03:53:16 +0200] "GET http://www.epochtimes.com/ HTTP/1.1" 200 756
222.79.51.217 - - [30/Jul/2020:03:53:16 +0200] "GET http://boxun.com/ HTTP/1.1" 200 756
123.145.24.197 - - [30/Jul/2020:03:53:16 +0200] "CONNECT www.baidu.com:443 HTTP/1.1" 405 899
171.36.140.77 - - [30/Jul/2020:03:53:17 +0200] "GET http://www.rfa.org/english/ HTTP/1.1" 404 966
223.12.74.86 - - [30/Jul/2020:03:53:17 +0200] "GET http://www.123cha.com/ HTTP/1.1" 200 756
171.12.10.146 - - [30/Jul/2020:03:53:20 +0200] "GET http://www.wujieliulan.com/ HTTP/1.1" 200 756
219.140.119.140 - - [30/Jul/2020:03:53:21 +0200] "CONNECT www.ipip.net:443 HTTP/1.1" 405 898
185.34.152.105 - - [30/Jul/2020:03:59:21 +0200] "GET / HTTP/1.1" 200 756
195.32.127.8 - - [30/Jul/2020:04:44:28 +0200] "GET / HTTP/1.1" 200 756
202.159.16.193 - - [30/Jul/2020:05:14:07 +0200] "GET / HTTP/1.1" 200 756
103.118.188.135 - - [30/Jul/2020:05:43:54 +0200] "GET /admin/login.asp HTTP/1.1" 400 923
195.54.160.21 - - [30/Jul/2020:06:19:59 +0200] "GET /?XDEBUG_SESSION_START=phpstorm HTTP/1.1" 200 756
195.54.160.21 - - [30/Jul/2020:06:19:59 +0200] "POST /api/jsonws/invoke HTTP/1.1" 404 969
195.54.160.21 - - [30/Jul/2020:06:19:59 +0200] "POST /vendor/phpunit/phpunit/src/Util/PHP/eval-stdin.php HTTP/1.1" 404 969
195.54.160.21 - - [30/Jul/2020:06:19:59 +0200] "GET /vendor/phpunit/phpunit/src/Util/PHP/eval-stdin.php HTTP/1.1" 404 969
195.54.160.21 - - [30/Jul/2020:06:19:59 +0200] "GET /solr/admin/info/system?wt=json HTTP/1.1" 404 969
195.54.160.21 - - [30/Jul/2020:06:19:59 +0200] "GET /index.php?s=/Index/\think\app/invokefunction&function=call_user_func_array&vars[0]=md5&vars[1][]=HelloThinkPHP HTTP/1.1" 404 969
195.54.160.21 - - [30/Jul/2020:06:19:59 +0200] "GET /?a=fetch&content=<php>die(&md5(HelloThinkCMF))</php> HTTP/1.1" 404 969
  
```

Figure 14 : activité anormale du serveur

Je constate que les attaques viennent du monde entier. Il est donc intéressant de mettre en place un filtre du type fail2ban sur le site qui bannit des IP pendant une à deux journées s'ils font un nombre minimal N réglable de tentatives ratées.



```

Terminal - damien@servperso:/home/damien
Fichier Édition Affichage Terminal Onglets Aide
2020-07-27 19:12:11,191 fail2ban.filter [444]: INFO [apache-noscript] Found 103.213.249.231 - 2020-07-27 19:12:11
2020-07-27 19:12:11,984 fail2ban.filter [444]: INFO [apache-noscript] Found 103.213.249.231 - 2020-07-27 19:12:11
2020-07-27 19:12:12,297 fail2ban.filter [444]: INFO [apache-noscript] Found 103.213.249.231 - 2020-07-27 19:12:12
2020-07-27 19:12:13,071 fail2ban.filter [444]: INFO [apache-noscript] Found 103.213.249.231 - 2020-07-27 19:12:13
2020-07-27 19:12:13,792 fail2ban.filter [444]: INFO [apache-noscript] Found 103.213.249.231 - 2020-07-27 19:12:13
2020-07-27 19:12:14,217 fail2ban.filter [444]: INFO [apache-noscript] Found 103.213.249.231 - 2020-07-27 19:12:14
2020-07-27 19:12:14,234 fail2ban.actions [444]: NOTICE [apache-noscript] Ban 103.213.249.231
2020-07-28 00:44:41,700 fail2ban.filter [444]: INFO [apache-noscript] Found 195.54.160.21 - 2020-07-28 00:44:41
2020-07-28 05:42:10,129 fail2ban.actions [444]: NOTICE [apache-noscript] Unban 47.254.144.218
2020-07-28 14:07:56,504 fail2ban.filter [444]: INFO [apache-noscript] Found 195.54.160.21 - 2020-07-28 14:07:56
2020-07-28 19:12:13,545 fail2ban.actions [444]: NOTICE [apache-noscript] Unban 103.213.249.231
2020-07-28 21:02:03,552 fail2ban.filter [444]: INFO [apache-auth] Found 121.223.12.94 - 2020-07-28 21:02:03
2020-07-29 03:49:33,883 fail2ban.filter [444]: INFO [apache-auth] Found 200.119.45.66 - 2020-07-29 03:49:33
2020-07-29 04:21:59,956 fail2ban.filter [444]: INFO [apache-noscript] Found 106.12.117.138 - 2020-07-29 04:21:59
2020-07-29 04:22:01,159 fail2ban.filter [444]: INFO [apache-noscript] Found 106.12.117.138 - 2020-07-29 04:22:01
2020-07-29 04:22:01,883 fail2ban.filter [444]: INFO [apache-noscript] Found 106.12.117.138 - 2020-07-29 04:22:01
2020-07-29 05:37:59,651 fail2ban.filter [444]: INFO [apache-noscript] Found 15.207.105.13 - 2020-07-29 05:37:59
2020-07-29 08:49:31,744 fail2ban.filter [444]: INFO [apache-noscript] Found 124.106.83.63 - 2020-07-29 08:49:31
2020-07-29 08:49:32,013 fail2ban.filter [444]: INFO [apache-noscript] Found 124.106.83.63 - 2020-07-29 08:49:31
2020-07-29 08:49:32,813 fail2ban.filter [444]: INFO [apache-noscript] Found 124.106.83.63 - 2020-07-29 08:49:32
2020-07-29 08:49:33,622 fail2ban.filter [444]: INFO [apache-noscript] Found 124.106.83.63 - 2020-07-29 08:49:33
2020-07-29 08:49:34,412 fail2ban.filter [444]: INFO [apache-noscript] Found 124.106.83.63 - 2020-07-29 08:49:34
2020-07-29 08:49:34,552 fail2ban.actions [444]: NOTICE [apache-noscript] Ban 124.106.83.63
2020-07-29 20:15:35,380 fail2ban.filter [444]: INFO [apache-noscript] Found 124.106.83.63 - 2020-07-29 20:15:35
  
```

Figure 15 : filtre Fail2ban et banissement d'IP

Le résultat obtenu figure 15 semble peu convaincant : en effet, Fail2ban ne détecte pas toutes les tentatives d'attaque. Je pense qu'il faut peut-être aller encore plus loin dans un cadre qui dépasse celui du stage en mettant une solution de type pare-feu sur le serveur en filtrant de manière plus efficace les connexions entrantes. Compte tenu de l'aspect sécuritaire inexistant dans ce stage, et du fait que le serveur ne manipule que les données du site, et n'héberge aucune donnée personnelle, je laisse la configuration telle quelle.

Conclusion

Au cours de ces huit semaines de stage, j'ai pu créer de toutes pièces un site web modélisant le modèle de durée de vie, en ayant pu appliquer les notions vues à l'école, puis également voir la possibilité d'exécuter des commandes système, comme par exemple des requêtes Python via PHP, et même aller encore plus loin en ayant traité le sujet de l'hébergement de ce site web. Le télétravail m'a également permis de m'investir encore plus dans ce stage et m'a permis d'être plus productif en étant soumis à un environnement connu (ma machine personnelle). Enfin, j'ai pu découvrir le monde de l'entreprise, avec toute l'organisation et les rendus qui en découlent.

Le site web créé peut encore être amélioré, notamment au sujet de l'amélioration des temps de traitements sans rogner sur la précision, et également sur la sécurité du site web en question, même si cet aspect est dans ce stage anecdotique.

Enfin, au cours de ce stage, j'ai proposé un site web qui est déjà utilisable tel quel, et permet par un client léger, de produire une analyse de données sans la nécessité d'installer des logiciels et langages de traitement. Il est alors une brique de base pour des futurs traitements encore plus évolués et pouvant être réalisés sur serveur et non en local.

Bibliographie

- [1] INRIA Grenoble Rhône-Alpes - *Seuil et p-valeur* - <https://mistis.inrialpes.fr/software/SMEL/cours/ts/node4.html>
- [2] M.A. Stephens - *Journal of the American Statistical Association* - vol 169, n°347, Septembre 1974, p.730-737
- [3] Documentation Scipy - <https://docs.scipy.org/doc/>
- [4] Bibmath - *Estimation et tests* - <http://www.bibmath.net/dico/index.php?action=rub&-quoi=50302>
- [5] Jean-Baptiste Apoung - *Optimisation sous contrainte en dimension finie* - https://www.imo.universite-paris-saclay.fr/~apoung/mywepage/MIM/2018/TP4_1718.pdf
- [6] Documentation Arch Linux - <https://wiki.archlinux.fr/LAMP>, <https://wiki.archlinux.org/index.php/PostgreSQL>
- [7] European Journal of Statistics and Probability - *On the sum of exponentially distributed random variables : a convolution approach* - vol 1, n°2, Décembre 2013, p.1-8
- [8] E. Morice - *Quelques modèles mathématiques de durée de vie* - Revue de statistique appliquée - vol 14, n°1, 1966, p.45-126

Annexe A : environnement de travail



Figure A : environnement de travail, éditeur de texte Atom

Annexe B : développement durable

B.1. Enjeux environnementaux et pressions

Étant donné le secteur de travail du stage qui est l'informatique, l'aspect environnemental est aussi intéressant à étudier : le CNAM propose d'ailleurs des formations concernant l'environnement et le développement durable, qui sont indépendantes. Néanmoins, parce que le CNAM est un établissement d'enseignement, les pressions sur l'environnement sont principalement matérielles et concernent notamment les dépenses énergétiques (électricité et confort climatique - chauffage ou climatisation), la gestion des déchets et les dépenses d'eau (robinets collectifs). On peut supposer que le bâtiment historique abritant le CNAM est une passoire thermique et que les dépenses énergétiques s'en retrouvent donc décuplées.

Par contre le télétravail ajoute des pressions supplémentaires : le serveur que j'ai mis en place pour héberger la page web tournant 24h/24, le canal mail et Teams pour communiquer générant des émissions de CO₂. Elles sont en fait communes au secteur informatique en général, et bien sûr, le stage s'y retrouve également.

B.2. Minimisation des pressions

Le contexte du stage, notamment le télétravail, fait qu'il est quasiment impossible d'avoir des détails sur une possible politique de réduction de ces pressions.

Concernant le télétravail, il permet d'éviter les déplacements entre mon domicile et le CNAM, même si ceux-ci auraient été effectués à vélo au vu de la proximité entre mon domicile et le CNAM (15 min à vélo, 4,2 km). J'ai également tenu compte des dépenses énergétiques concernant la page web que j'ai développée pour le stage : en effet, j'ai réduit au maximum les requêtes Javascript pour éviter de trop solliciter le processeur. Concernant le serveur mis en place pour le stage, il s'agit comme décrit à la partie 6.1. ([p.20](#)) d'un ordinateur portable fonctionnant sur batterie ou secteur et possédant un processeur à faible consommation. En effet, la puissance totale instantanée de l'ordinateur oscille entre 2 W à vide et 7 W à pleine charge.

On peut calculer la consommation du serveur durant les deux mois de stage : en supposant que le serveur tourne très majoritairement à vide et que les traitements du site et le système en fond n'utilisent que ponctuellement le processeur à pleine charge, on peut estimer la puissance moyenne du serveur à 4 W. Cela représente donc sur les deux

mois de stage une consommation de **5,76 kWh**. En comparaison, un PC de bureau classique consomme sur les deux mois de stage 90 kWh (puissance de 150 W pendant 60 jours à 10h/jour). Ainsi, le serveur mis en place n'a quasiment pas d'impact sur la dépense énergétique globale et contribue à améliorer significativement le déroulement du stage en télétravail.

D'ailleurs, la fermeture du CNAM a également pu faire réduire drastiquement les dépenses énergétiques (que ce soit climatisation ou électricité, et qui sont à une autre échelle à contrario des particuliers) ce qui est une des conséquences positives du confinement ; néanmoins, l'utilisation massive de mails et de canaux de visioconférence comme Teams à la place de discussions en présentiel a fait augmenter les émissions de CO₂ : à raison de 4 mails par semaine sur les deux mois de stage et des différents mails administratifs, on arrive à une émission de **400 g de CO₂** (10 g de CO₂ par mail environ), sans compter l'empreinte carbone des visioconférences, qui est très difficile à estimer. Cela contrebalance la réduction des dépenses énergétiques. En comparaison, un trajet simple en métro vers le CNAM aboutit à une émission de **15,96 g de CO₂**. (3,8 g/km d'émission de CO₂)

En conclusion, un stage en présentiel permet de supprimer ces dépenses et permet de réduire l'empreinte carbone du stage, mais ferait augmenter la dépense énergétique globale. Tout est une question d'équilibre, mais il y a une certitude : l'empreinte carbone du stage en télétravail est loin d'être nulle et ce point est à améliorer dans le futur.